Exam 2 CS-GY 6033 INET Fall 2024
Noember 18th 2024

Instructions:

Scheduling:

- The exam runs from 6pm to 9pm EST on November 18th 2024. Your exam is run through Gradescope, and will available to download at 5:50pm. The exam is to be completed by 9:00pm. Gradescope stops accepting uploads at 9:15pm. There are absolutely no late uploads accepted by the system after that time. The time to complete the exam will depend on your preparation: a prepared student could finish it in less than two hours, an ill-prepared student might take up to three hours. It is your responsibility to allow time for uploading your exam.

Format:

- The exam consists of a total of 80 points. The remaining 20 points are for the online quiz.
- **Submission Penalty: if you do not submit your exam on Gradescope, or do not properly assign your pages, you receive a deduction of 3 points on your exam.**
- You may write your solutions directly on the downloaded exam paper, *or* in your own format. You are responsible for providing clear and legible solutions to the problems. Your exam must be resubmitted into Gradescope electronically. Ensure that you know how to quickly upload any handwritten material. This is entirely the student's responsibility. You may assign your pages after the deadline.

Questions during the exam:

- There is a ZOOM session for questions that will be open during the entire course of the exam (microphones OFF). You may ask questions with private chat during the exam. Any announcements made by the instructor during the exam will be made over ZOOM and also be email. **It is the student's responsibility to stay connected (either by ZOOM or email) during the exam.**

Rules:

- This exam is a **take-home exam**. You may use **only** the resources from the online class (any material on NYU classes for this course) and any type of calculator (although it is not needed).
- Your work must be entirely your own. It is **forbidden to discuss any work with *any* other person**. Furthermore, your work must be done without using internet searches (although this is completely unhelpful for this exam). Any breach of academic honesty will be handled in accordance with the *Student Code of Conduct*, (a copy of which is provided), and in this particular case, taken very seriously.
- You are asked to **read** the attached Student Code of Conduct Section III subsections A,B,C,D,E and **sign** below to acknowledge that you aware of the policy. Once signed, a copy of this page must be uploaded with your exam.

I acknowledge that my submitted Exam work is entirely my own. I have read and am in accordance with the Student Code of Conduct policy of NYU Tandon and fully accept the consequences of breaching the above instructions.

Name: ______________________________

Signature: ______________________________

Instructions

Read the following questions carefully. Recall that you may use algorithms from class. For example, you do not need to re-describe from scratch algorithms like inserting into a BST, Inorder Traversal, or using the Rank Algorithm.

Your written exam has a total of 80 points.

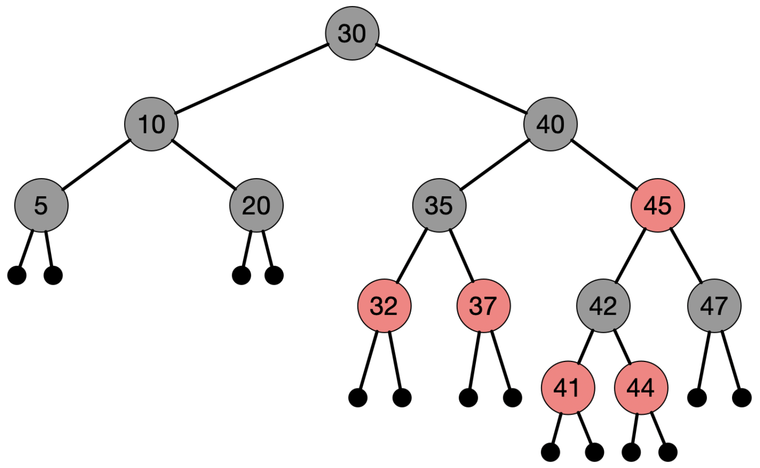
ZOOM LINK: https://nyu.zoom.us/j/96190643479

MICROPHONES OFF!

Question 1

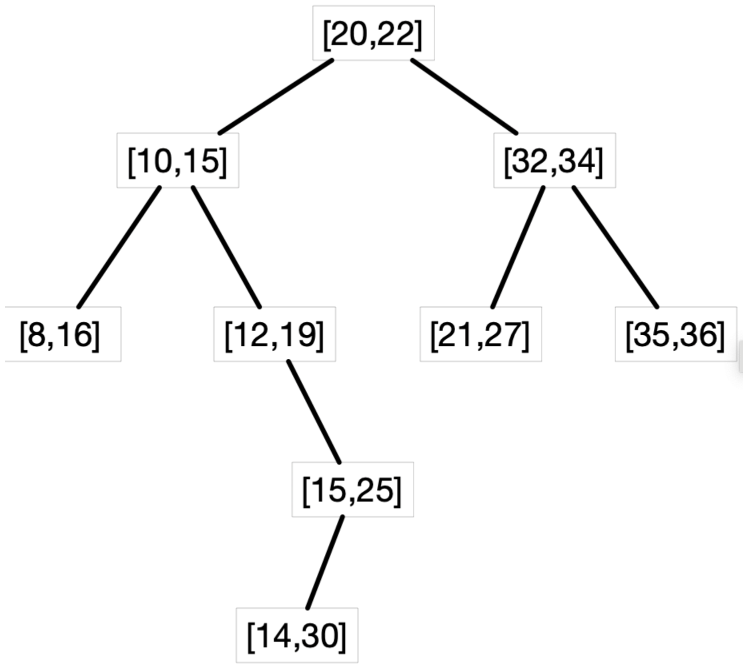
(a) 6 points The drawing below represents a valid red-black tree.

- Show how to insert the new node 43. You must show both the initial insertion, and any changes made by RB-repair.
- The tree now has 14 nodes. What is the **maximum** number of nodes that can be **added** before the black heigh increases?



(b) 6 points An interval tree is drawnn below. Answer each of the following:

1. Is it possible to assign black/red colors to the tree and add NIL nodes so that the tree is implemented as a Red-black tree? (Do not change its shape).
2. Add the attributes $x.max$ to each node
3. Suppose we carry out the INTERVAL-SEARCH(i) algorithm from class, where $i = [23, 24]$. Show which node is returned.
4. Given an example of an interval i for which INTERVAl-SEARCH(i) returns node $[21, 27]$, or explain why it is impossible.



Question 2

4 points

(a) Give an example of a BST on 10 nodes such that the Inorder and Postorder traversal produce the same output, or explain that it is not possible

(b) Give an example of a BST on 10 nodes such that the Preorder and Postorder traversal produce the same output, or explain that it is not possible

Question 3

4 points

Let T be the root of a BST augmented with the attribute $x.leaves$, which is the number of leaves in the subtree rooted at x. Suppose T is not initially augmented with this value. Write the pseudo-code for a procedure called SetLeaves(T) which correctly sets the attribute $x.leaves$ for all nodes in the tree. Justify the runtime of $O(n)$.

Question 4

6 points

Consider the above BST augmented with $x.leaves$. Describe how to update the **TreeInsert** algorithm from class so that the attributes $x.leaves$ are correctly updated after an insert of node z into the tree T. You do not need to explicitly provide the pseudo-code for the new version of TreeInsert. Instead, you must carefully describe how the procedure is updated, step by step, and justify the runtime of $O(h)$.

Question 5

6 points

Let T be a Red-Black tree, which is also **complete** (recall the definition of complete is that every level is full, except perhaps the last, where the last level is filled in from left to right). A student claims that the RBT is colored in such a way that all levels are colored black, except the last level. Your job is to write the pseudo-code for an algorithm called CheckColoring(T) which takes in a reference to the red-black tree T, and returns TRUE if indeed the tree is colored all black except the last level (which is red). Otherwise the procedure returns FALSE. You may call another algorithm from within CheckColoring if you find that helpful. Justify the runtime of $O(n)$.

Question 6

Let T be a binary search tree that stores information on exam grades for students in the Algorithms class. Each tree node x contains the following attributes:

x.grade: Grade of the student, used as the *key* of the BST
x.prereq: Grade the student x achieved in the prerequisite course
x.credits: Number of credits currently achieved by student x
x.maxcredits Maximum number of credits achieved by all students in the subtree rooted at x
x.PMax: Maximum of the prerequisite course grades for all students in the subtree rooted at x
x.size:. Number of nodes in the subtree rooted at x

(a) 2 points Describe how to implement the above BST such that inserts and deletes can be carried out in time $O(\log n)$.

(b) 4 points Write the pseudo-code for a recursive algorithm called MaxCredits(T), which returns the **maximum number of credits** achieved by any student in T with an Algo Exam grade of 80 or more. Justify the runtime of $O(\log n)$

(c) 4 points Write the pseudo-code for a recursive algorithm called MaxPregrade(T), which returns the **maximum grade** achieved in the prerequisite course by any student in T with an Algo Exam grade under 60. Justify the runtime of $O(\log n)$.

(d) 4 points Consider the student in the class with the highest grade in the pre-requisite course. Determine how many students achieved a grade *higher* than this student on the Algo Exam. Write the pseudo-code for your procedure, called AboveBest(T). Justify the runtime of $O(\log n)$.

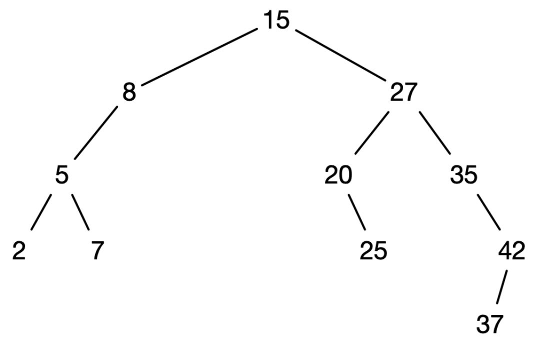
Question 7

4 points

Consider T which is a reference to the root node of a binary search tree. Let PrintDepth(T, i) be a procedure that prints the nodes in T at depth i. The pseudo-code for this procedure is given below:

```
PrintDepth($T, i$)
    if $i = 0$ print $T.key$
    else
        PrintDepth($T.left, i - 1$)
        PrintDepth($T.right, i - 1$)
```

Using the procedure above, your job is to write a procedure that prints the nodes of a BST level by level. In the example below, the output should be $15, 8, 27, 5, 20, 35, 2, 7, 25, 42, 37$.



Question 8: DP WARM UP

8 points Suppose we are given a set of weights $w[1, 2, \ldots, n]$ with their associated prices $p[1, 2, \ldots, n]$ and a target total weight T. The goal is to select a set of weights whose **total sum is exactly equal to T,** such that we **minimise the total price of the selected weights**.

Your Job:

Provide a DP solution for this problem.

You MUST use the following DP table: $M[0, 1, \ldots, n, 0, 1, \ldots, T]$ where $M[i, j]$ is the minimum possible cost of achieving **exactly** weight j using weights selected from $0, 1, 2...i$. If achieving exactly weight j is impossible, you may set entry $M[i, j]$ to an any flag value that works in your solution.

You must include:

- justify how you initialise the table
- the relationship you use to fill up the table
- the pseudo-code
- the runtime

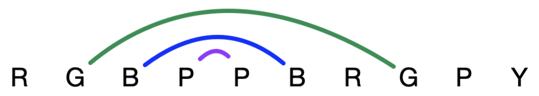
You may use the MaxValueSet problem from class as reference, where the original pseudo-code is copied below.

```
for $j = 0$ to $T$ set $V[0, j] = 0$
for $i = 0$ to $n$ set $V[i, 0] = 0$
for $i = 1$ to $n$
    for $j = 1$ to $T$
        if $w[i] \leq j$
            $V[i, j] = \max\{v[i] + V[i-1, j - w[i]], V[i-1, j]\}$
        else $V[i, j] = V[i-1, j]$
return $V[n, T]$
```

Question 9: DP WARM UP

6 points

Suppose we have a sequence of characters in the array $C[1, 2, \ldots, n]$ where each character represents a **color**. The characters are selected from the set $\{R, B, G, Y, P\}$. An example of such a sequence is $RBBGYPRGBYPYPY$. Using these characters, we would like to draw a **rainbow**, where an arc of a certain color can be made be connecting two of the same color. For a proper rainbow to be drawn, colored arcs cannot cross each other! An example of a rainbow is shown below:



Arcs of different colors have different values. Yellow arcs are worth \$100. Blue arcs are worth \$200. Green arcs are worth \$300. Pink arcs are worth \$400. Red arcs are worth \$500.

Your Job:

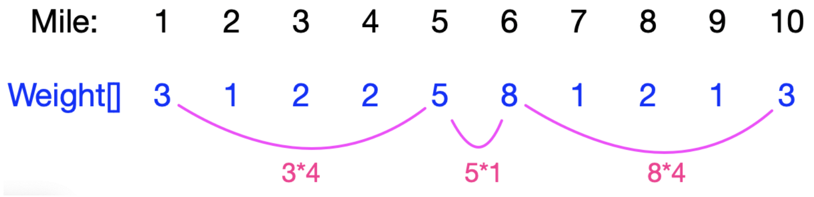
Update the relevant DP problem from class so that it returns the **maximum value** of a rainbow that is drawn using from the characters in $C[1, 2, \ldots, n]$.

You must properly define the DP table, explain the initialisation, justify how you fill in the entries, provide the pseudo-code, clearly show which value is returned, and justify the runtime.

Question 10

6 points

Suppose we have a hiking trail that goes from mile marker 1 to mile marker n. At each mile marker there is **ONE rock** with a certain weight. If we pick up the rock at mile marker i and carry it to mile marker j, we get paid an amount which depends on the weight and how far we carried it, where the payment is $(j - i) \times (weight)$. An example is given below, where our total profit is 49.



Suppose you are given as input the array $W[1, 2, \ldots, n]$, where $W[i]$ represents the weight of the rock at marker i. The goal is to determine the maximum amount of money that can be made. Note that you can only walk forwards on the trail, and that you can only carry one rock at a time! You are allowed to drop a rock and pick up a rock at the same mile marker, or you may drop a rock and continue walking with nothing.

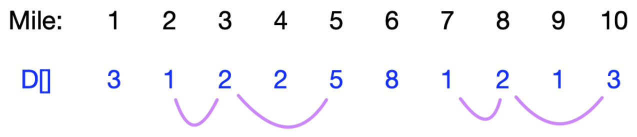
Suppose a student provides a *greedy* approach to solving this problem which works as follows:
-Pick up the rock at mile 1
-Walk until we find a larger rock, and then put the current rock down and trade up for the heavier rock.
-Continue until we get to the last mile marker.

Your Job:
Determine if this greedy approach produces the optimal solution. You must either give an **example** showing that this approach does **not** produce the best result, OR, you must explain why this greedy approach always produces the maximum amount of money.

Question 11

12 points

Consider another hiking trail that goes from mile marker 1 to mile marker n. At each marker, there is again ONE rock. Each rock must now be carried a certain number of miles. The input to the problem is $D[1, 2, \ldots, n]$ where $D[i]$ is the **distance** that we much carry rock i. For example, if $D[1] = 4$, the rock at marker 1 must be carried at least 4 miles. At each mile marker, we have the choice of picking up the rock, or continue walking with no rock. We cannot carry more than one rock at a time, but we are allowed to drop a rock, and pick a new one up at the same mile. Any rock that we pick up MUST be carried the minimum number of miles. In the example below, the rock pick ups and drop offs show that we are able to transport four rocks.



Your job:

Write a DP solution that solves the problem of finding the **maximum** number of rocks that can be successfully transported during the hike. Be sure to include a properly defined DP table, describe the initialization, how the table is filled up, the pseudo-code, and the runtime.

BONUS: 5 points Update the above solution so that you also print out the optimal solution: this consists of the the mile markers where you pick up rocks.